UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/778,424	02/07/2001	Joseph C.H. Park	03226.037001; P5009	6879

32615 7590 03/29/2004

OSHA NOVAK & MAY L.L.P./SUN
1221 MCKINNEY, SUITE 2800
HOUSTON, TX 77010

EXAMINER	
VU, TUAN A	

ART UNIT	PAPER NUMBER
2124	5

DATE MAILED: 03/29/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/778,424	PARK, JOSEPH C.H.	
	Examiner	Art Unit	
	Tuan A Vu	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *07 February 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-23* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-23* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *02/07/2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is responsive to the application filed February 7, 2001.

Claims 1-23 have been submitted for examination.

Claim Objections

2. Claim 22 is objected to because of the following informalities: It appears that there is typo error in element recited as 'method of claim 22' (line 1). It should be corrected to be 'method of claim 21'.

Appropriate correction is required.

Claim Rejections - 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

4. Claims 8, 9, 18, 21 and 23 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

The Federal Circuit has recently applied the practical application test in determining whether the claimed subject matter is statutory under 35 U.S.C. § 101. The practical application test requires that a " useful, concrete, and tangible result" be accomplished. An "abstract idea" when practically applied is eligible for a patent. As a consequence, an invention, which is eligible for patenting under 35 U.S.C. § 101, is in the "useful arts" when it is a machine, manufacture, process or composition of matter, which produces a concrete, tangible, and useful result. The test for practical application is thus to determine whether the claimed invention produces a "useful, concrete and tangible result".

Claims 8 and 9 recite a rule-based system and method, respectively, for 'optimizing predicated code', and comprising 3 action steps performing actions defining, testing and assigning. But there is no sufficient description as to how those steps can cumulatively accomplish the intended subject matter recited as 'optimizing predicated code', or can clearly lead to a tangible and concrete result called for by such 'optimizing'. The processor element can be viewed as potentially supporting the subject matter intended by the preamble; however, the

steps of defining, testing, assigning, which nominally are to describe how such subject matter is implemented, do not amount to a practical and useful end result for they all appear to be leading toward a non-practical use, or nothing suggestive that an optimization result has been tangibly or concretely accomplished (e.g. the last step of assigning remains abstract when such assigning does not have a concrete impact of the optimization of code). Absent a tangible and useful result, the claims are directed to a non-statutory subject matter because they fail the requirements of a practical application test.

Claims 18 and 21 recite a technique and a method, respectively, for 'supporting predicated execution without explicit hardware', and comprising one action of 'implementing'. This action description does not provide sufficient elements as to accomplish a tangible result as required by the practical test requirement, i.e. it is further unclear how the recited step of 'implementing a test branch instruction' does usefully accomplish the subject matter recited as 'supporting predicated execution'. Further, the action recited as 'implementing' amounts to a broad action without specificity as to how it can accomplish a practical result or concrete result, hence remains abstract and therefore fails to satisfy the practical test. Absent a concrete, useful, and tangible result, the claims are directed to a non-statutory subject matter because they fail the requirements of a practical application test.

Claim 23 recites an apparatus for 'supporting predicated execution without explicit hardware' and means for 2 step actions of 'implementing' and 'eliminating'. As mentioned in the rejection of claims 8 and 9, the steps elements recited to supposedly implement the subject matter of the preamble do not amount to yielding a concrete and tangible result as to support the 'predicated execution'.. As recited, it is not explicit as to how 'implementing a test branch' and

'eliminating predicates' can lead to a practical result in the context of usefully 'supporting predicated execution without explicit predicate hardware', i.e. they lead to a non-practical use for no concrete result is generated therefrom. Absent a concrete, useful, and tangible result, the claims are directed to a non-statutory subject matter because they fail the requirements of a practical application test.

The claims dependent from claims 8, 9, 18, 21, and 23 are also rejected for being dependent on a rejected base claim.

Claim Rejections - 35 USC § 112

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. Claims 8, 9, 18, and 21 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claims 8 and 9 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter.

The term "based on a result" in claims 8 and 9 (line 7, 6, respectively) is a relative term which renders the claim indefinite. The term "based on" and the term 'a result' are not defined by the claim; the specification does not provide a standard for ascertaining the requisite degree on how an action to 'based on a result' is performed and such according to what criteria, and one of ordinary skill in the art would not be reasonably apprised of the scope of the invention. The examiner will interpret this as if the limitation 'based on a result of the test' were 'as a result of said test'.

Claims 18 and 21 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for reciting a step which amounts to a omnibus limitation. The only step of 'implementing' is not defining the details as to how 'implementing' is to support the subject matter, such lack of definition amounting to a misconnection between the body of the claim and the purported subject matter of the preamble.

The dependent claims from claims 9, 18, and 21 are also rejected for being dependent on a rejected base claims.

Claim Rejections - 35 USC § 102

7. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

8. Claims 1, 7-10, 16, and 18-23 are rejected under 35 U.S.C. 102(b) as being anticipated by Intel® IA-64 "Architecture Software Developer's Manual", Jan. 2000, (hereinafter IA-64).

As per claim 1, IA-64 discloses an extensible rule-based technique for optimizing predicated code comprising if-converting an abstract internal representation (e.g. §10.2.3.1, pg. 10-3 to pg. 10-4 – Note: analysis of instruction sequence by a compiler for branch prediction optimization implicitly discloses analysis of internal representation of program code); mapping the if-conversion to a machine representation (assembler - §10.2.3.1, pg. 10-3,4).

As per claim 7, IA-64 discloses optimizing of machine representation (see §10.2.3-10.2.4, pgs. 10-6 -10-8)

As per claim 8, IA-64 discloses an extensible rule-based system for optimizing predicated code comprising a processor (Note: this is inherent to architecture IA-64); and a instruction for defining predicates; testing a branch instruction (e.g. *set predicate reg cmp.ne p1, p0= r4, 0* -- § 10.2.3.1); and assigning a defined predicate to the branch instruction as a result of said test (e.g. *p1, p2* - §10.2.3.1, pg. 10-3 to pg. 10-4).

As per claim 9, refer to claim 8; further IA-64 discloses selectively assigning a defined predicate to the branch instruction as a result of said test (e.g. *p1, p2* - §10.2.3.1, pg. 10-3 to pg. 10-4; §10.2.4 – Note: choosing to set predicates based on guarantee that no conflict or inter/overlapping resources incur is equivalent to selectively applying predicate assignment of branch instruction).

As per claim 10, this apparatus claim corresponds to claim 1 and is rejected with the rejection as set forth therein.

As per claim 16, this apparatus claim corresponds to claim 7, and is rejected with the rejection as set forth therein.

As per claims 18 and 21, IA-64 discloses a technique/method of supporting predicated execution without explicit hardware, comprising implementing a test branch instruction (§10.2.3.1, pg. 10-3 to pg. 10-4).

As per claim 19, IA-64 discloses converting a branching condition based on codes to Boolean in a general register to that a full logical instruction set can be used to produce optimal code (*set predicate reg cmp.ne p1, p0= r4, 0* -- § 10.2.3.1).

As per claim 20, IA-64 discloses a system of supporting predicated execution without explicit hardware, comprising

a processor; and

instructions for converting a branching condition based on codes to Boolean in a general register to that a full logical instruction set can be used to produce optimal code (*set predicate reg cmp.ne p1, p0= r4, 0* -- § 10.2.3.1); and

guarding a set of instructions (e.g. *p1, p2* - §10.2.3.1, pg. 10-3 to pg. 10-4; §10.2.4) unsuitable to speculate enclosed by a branch (Note: providing predicate register test to avoid go-ahead execution of an assumingly correct predicted path -- if-code without predication -- reads on guarding set of instructions to prevent fault from otherwise speculatively determined control flow, i.e. unsuitable speculated execution).

As per claim 22, this apparatus claim corresponds to claim 19, and is rejected with the rejection as set forth therein.

As per claim 23, IA-64 discloses an apparatus of supporting predicated execution without explicit hardware, comprising

means for implementing a test branch instruction (§10.2.3.1, pg. 10-3 to pg. 10-4); and

means for elimination of predicates from mapped if-conversion (e.g. §10.2.4, pgs. 10-6-8); hence has disclosed eliminating predicates using the implemented test branch instruction from such if-conversion.

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 2, 3, 4, 6, 11-13, and 15 are rejected under 35 U.S.C. 103(a) as being unpatentable over Intel® IA-64 "Architecture Software Developer's Manual", Jan. 2000 (IA-64), as applied to claims 1 and 10 from above, in view of Hwu et al., "A Framework for Balancing Control Flow and Predication", 1997 (hereinafter Hwu).

As per claim 2, IA-64 suggests not to use predicates under some conditions when applying the if-conversion/predication analysis (e.g. §10.2.4, pgs. 10-6-8). Hwu, in a method using hyperblock analysis to institute if-conversion analogous to the if-converting optimization by IA-64, also discloses the pitfalls of using predication when hardware resources, interference or repeated use of same resources and scheduling time frame are becoming a prohibitive factor (pg. 93, left col, pg. line 5 to pg. 95, left col.), and ways to un-execute predication settings (associated with hyperblock analysis during the if-conversion) by applying the partial reverse if-conversion when the code to execute from the if-conversion is not desirable, or that hyperblock control flow lead to penalties (e.g. pg. 96, right col. to pg. 97, left col.). It would have been obvious for one of ordinary skill in the art at the time the invention was made to apply the substituting of predicate instructions during if-conversion as analyzed and performed by Hwu and apply it to the proposed considerations by IA-64 for the reasons both mentioned by Hwu and IA-64 which are worsening the delay and resource usage during scheduling and execution of predicated code.

As per claim 3, IA-64 does not expressly disclose eliminating a predicate defining instruction by interpretation but teaches not to use predicate instructions under unbalanced execution paths and overlapping resources (re claim 2). Hwu, in a method using hyperblock analysis to institute if-conversion analogous to the if-converting optimization by IA-64, also

discloses the pitfalls of using predication (re claim 2) and suggest ways to un-execute predication settings (associated with hyperblock analysis during the if-conversion) by applying the partial reverse if-conversion when the code to execute from the if-conversion is not desirable, or that hyperblock control flow lead to penalties (e.g. pg. 96, right col. to pg. 97, left col.); but does not explicitly teach using interpretation to replace predication execution as predication execution would incur penalties. In view of IA-64 and Hwu's teachings to apply reverse if-conversion at the late stage of code scheduling, the if-conversion associated with predicate guarded instructions is noted as aiming at optimizing resources from branch mispredicted paths and delays resulting therefrom, the idea of optimizing at run-time is suggested, i.e. optimizing about exactly before runtime by many compilers (e.g. JIT) was strongly suggested by the architecture taught by IA-64. Further, official notice is taken that one method to alleviate runtime resource is to use interpretation of code when just-in-time optimization would have been useful any further was a known-concept in the art of runtime optimization, e.g. the well-known JIT compiler and Java optimizing virtual machine, at the time the invention was made. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the interpretation of code to substitute for some execution provided by the reverse if-conversion as suggested by Hwu (in combination with IA-64) to avert unsafe predicate execution. One of ordinary skill in the art would be motivated to do this because, when the code to translate is a language which can be used by an just-in-time interpreter as mentioned above, enabling interpretation of code after if-converting as suggested by Hwu (in combination with IA-64) does not further complicate compilation resources when every possible optimization process would have been exhausted and that only size-simplified, branch-free and fault-free instructions

are left to execute just as evidenced in the teachings from the notice from above and in conjunction with the analysis and decision making in Hwu's algorithm for reverse if-converting hyperblocks (see Hwu: pg. 97-100).

As per claim 4, IA-64 does not specify eliminating a guard predicate of a safe instruction by speculation but discloses using both speculation and predication (e.g. § 2.6, pg. 2-4 to pg. 2-5; § 8.4-5, pg. 8-4 to pg. 8-6); and discloses how using predication can be inappropriate when overlapping memory resources are identified (re claim 2). Hwu, as in claim 3, further discloses applying an analysis to ensure that a predication is resolved and free of penalty (resources, cycle count, interaction, dangling latencies - pg. 97-100) by applying a reverse if-conversion to optimize resource usage at scheduling time. Official notice is taken that using speculation when it is determined that a code is free of ambiguous control flow or data dependency as suggested by IA-64 (in combination with Hwu) was a known-concept at the time the invention was made. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the speculation of code to substitute for some predicated execution as suggested by Hwu (in combination with IA-64) to avert unsafe predicate execution when it is determined that there is no ambiguous dependency of control or data flow as taught by Official notice from above and by Hwu's techniques. One of ordinary skill in the art would be motivated to do this because enabling speculation of code after the predicates resources dependency analysis as suggested by Hwu's correct scheduling (in combination with IA-64/Official notice) further alleviates execution resources from correctly predicting of data/control flow just as intended by the optimization proposed by IA-64 (combined with Hwu's teachings), notably when predicate implementation tends to increase size of code (re IA-64 – pg. 10-6 to pg. 10-8).

As per claim 6, IA-64 does not disclose using reverse if-conversion but this limitation would have been obvious in view of IA-64's teachings about the pitfalls in predication (re claim 2) and the rationale using Hwu's partial reverse if-conversion method.

As per claim 11, this apparatus claim corresponds to claim 2 and is rejected with the rejection as set forth therein.

As per claims 12, 13, 15, these claims correspond to claims 3, 4, 6 respectively, and are rejected with the rejections as set forth therein.

11. Claims 5, 14, and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Intel® IA-64 "Architecture Software Developer's Manual", Jan. 2000 (IA-64), as applied to claim1 (for claim 5), 10 (for claim 14) above, in view of Hwu et al., "A Framework for Balancing Control Flow and Predication", 1997; and further in view of Schlansker et al., USPN: 5,999,738(hereinafter Schlansker).

As per claim 5, IA-64 (combined with Hwu) does not disclose eliminating a guarding predicate of an unsafe instruction by compensation. IA-64 and Hwu, as in claim 3, discloses using analysis to determine resources dependency, cycle counts and interdependent scheduling (re claim 3-4) for averting using of penalty-prone or inefficient predicate-guarded block of instructions. Schlansker, in a method to selectively execute fully or partially resolved predications analogous to the partial reverse if-conversion by Hwu, discloses compensation code to stand for predicate code when the latter undergoes resolution as to whether it can fall-through or not (Fig. 10); hence has suggested substituting predicate instruction in case such predicate result in non fall-through resolution. Hence, it would have been obvious for one of ordinary skill in the art at the time the invention was made to implement the use of compensation code to

substitute for some predicate guard resolution leading to unsafe code as suggested by Hwu (in combination with IA-64) to avert unsafe predicate execution when it is determined that there potential unresolved predicated control flow can result in a penalty as taught by Schlansker. One of ordinary skill in the art would be motivated to do this because enabling compensation code to cover for predicate ill-selected paths as suggested by Schlansker's provision further alleviates execution resources from having to remedy for mispredicted path or paths resulting in wrong direction as originally intended by the optimization proposed by IA-64 (combined with Hwu's teachings), notably when predicate implementation tends to increase size of code (re IA-64 – pg. 10-6 to pg. 10-8).

As per claim 14, this claim corresponds to claim 5, and is rejected with the rejection as set forth therein.

As per claim 17, IA-64 discloses an extensible rule-based technique for optimizing predicated code comprising if-converting an abstract internal representation (e.g. §10.2.3.1, pg. 10-3 to pg. 10-4); mapping the if-conversion to a machine representation (assembler - §10.2.3.1, pg. 10-3 to pg. 10-4); eliminating of predicates from mapped if-conversion (e.g. §10.2.4, pgs.10-6-8).

But IA-64 does not expressly disclose eliminating a predicate by interpretation as recited in claim 3, by speculation as recited in claim 4, by compensation as recited in claim 5, by reverse if-conversion as recited in claim 6; but these limitations have been addressed with the corresponding rejections as set forth therein, respectively.

Conclusion

12. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Warter et al., "Reverse If-Conversion", 1993, ACM-SIGPLAN-6/93, disclosing intermediate representation and anti-dependency.

August et al., "The Partial Reverse If-Conversion Framework for Balancing Control Flow and Predication", May 17, 1999, Hewlett-Packard, Palo Alto, CA, disclosing hyperblock heuristics and dynamic reverse if-conversion.

U.S. Pat No. 6,513,109 to Gschwind et al., disclosing computing predicted value in predicate instructions.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Tuan A Vu whose telephone number is (703)305-7207. The examiner can normally be reached on 8AM-4:30PM/Mon-Fri.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (703)305-9662.

Any response to this action should be mailed to:

Commissioner of Patents and Trademarks

Washington, D.C. 20231

or faxed to:

(703) 872-9306 (for formal communications intended for entry)

or: (703) 746-8734 (for informal or draft communications, please label "PROPOSED" or "DRAFT" – please consult Examiner before use)

Hand-delivered responses should be brought to Crystal Park II, 2121 Crystal Drive, Arlington. VA. , 22202. 4th Floor(Receptionist).

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

VAT
March 21, 2004

KAKALI CHAKI
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100